A Unified Storage and Deployment Model for the Emulab Network Testbed

Raghuveer Pullakandam, Mike Hibler and Eric Eide
Flux Research Group, School of Computing
THE UNIVERSITY OF UTAH

Motivation

Problem

It is difficult for **Emulab** users to customize disk images for realistic network experiments

- Monolithic disk images
- Lack of variety
- Multiple imaging mechanisms
- Lack of flexibility

Goals

- Provide flexibility and variety using a **unified**, **efficient** and **scalable** mechanism
- **Flexibility** for creating custom client environments
- **Variety** of OSes and apps to promote realism

Approach

A new **model for encapsulating** disk contents to provide greater flexibility and control of the client environment

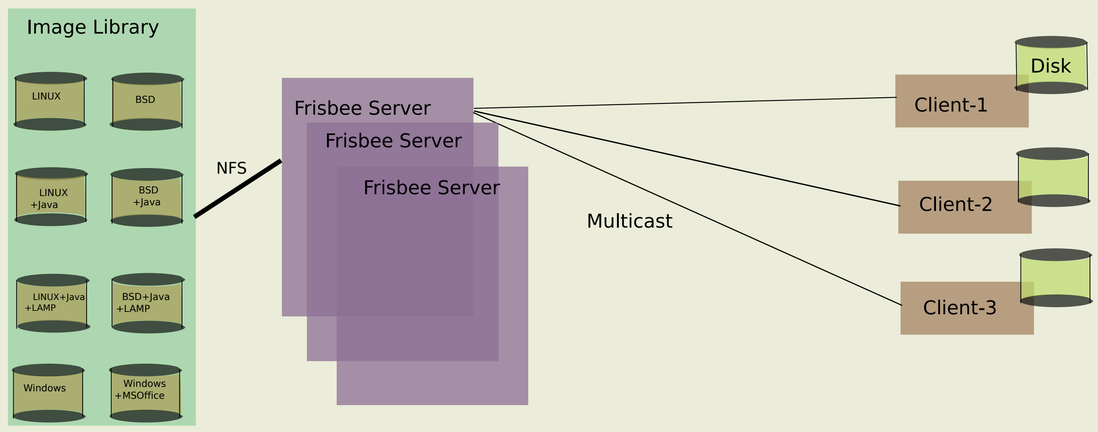
Efficient and Flexible Storage

- Server-side de-duplication
- Versioned images
- Dynamic image creation
- Greater user accessibility

Scalable Deployment

- Reduced network traffic
- Reduced image redundancy
- Leverage client-side storage
- Flexible image specification
- User-contributed delta images
- Support dynamic client setup

Current Frisbee Model



Frisbee efficiently distributes disk images and configures nodes in the Emulab Network Testbed

- Server partitions a disk image into independent chunks
- Server multicasts chunks to the clients

Issues & Limitations

- Designed for image distribution, not image management
- Redundant image content
- No support for dynamic image creation
- Emphasis on keeping clients busy, not server-side optimization
- Scalable in number of hosts, not number of images

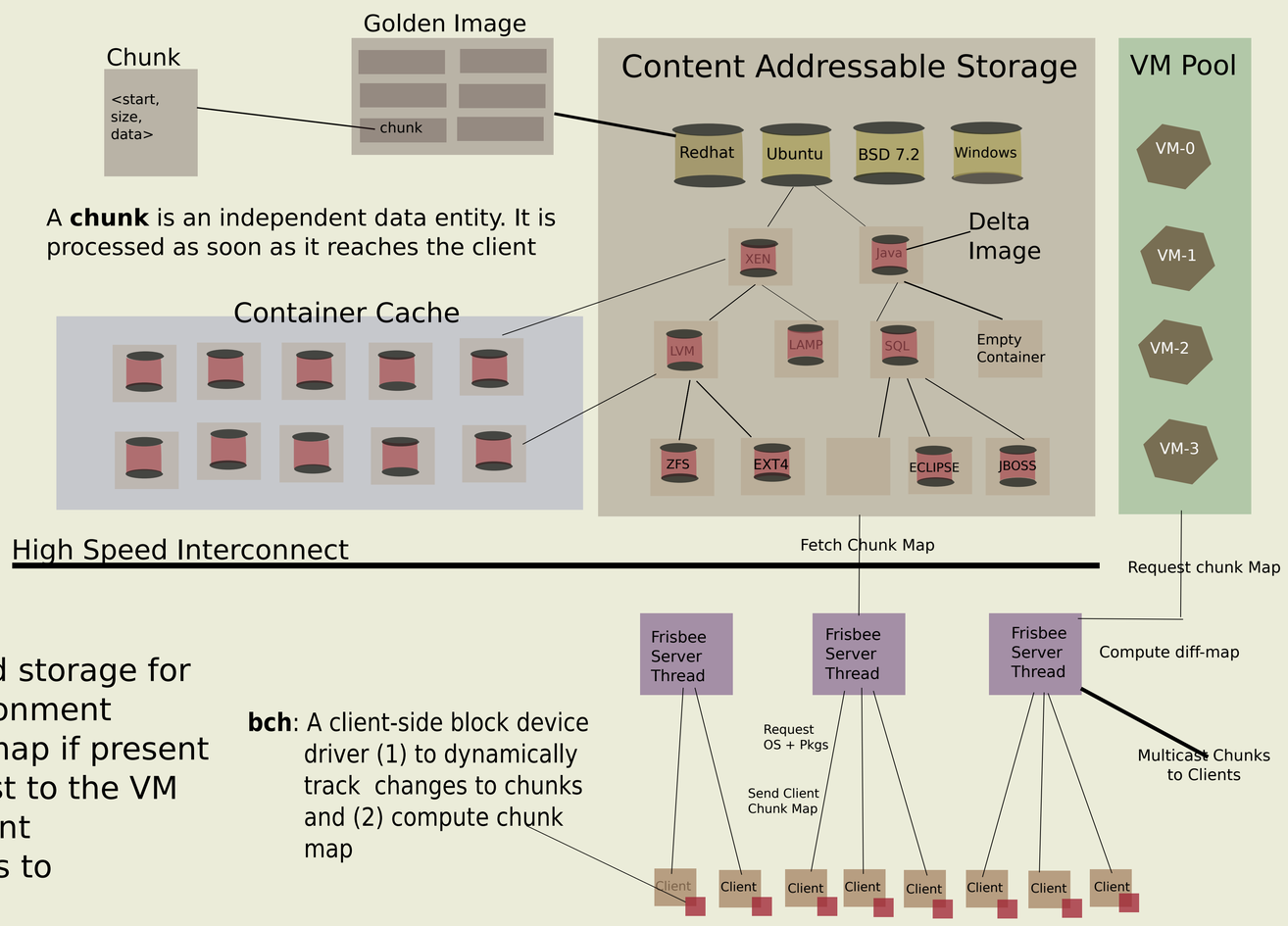
New Proposed Frisbee Model

Golden Image

- An immutable disk image
- Typically a base OS

Container

- Abstraction associated with an application



Frisbee Server

- Queries backend storage for requested environment
- Fetches chunk map if present
- Forwards request to the VM pool if not present
- Identifies chunks to multicast

bch: A client-side block device driver (1) to dynamically track changes to chunks and (2) compute chunk map

Delta Image

- Usually corresponds to a package/patch or a OS revision
- Applied on top of a golden image

VM Pool

- A pool of pre-configured virtual machines
- Dynamic VM allocation
- Loads disk image that is closest to requested configuration
- Installs required environment
- Sends updated chunk map to Frisbee server

Chunk Map

- Description of chunks in a disk image

Diff Map

- Description of chunks that need to be multicast

Applications

Contextualization

- Fast setup of a distributed system by imparting contextual information into the data chunks thus avoiding the need for client-side agent scripts
- Single chunk dispatched to multiple clients thus saving bandwidth

Fast **reimaging** of client nodes in a testbed

- Reimaging client nodes requires replacing only modified chunks
- Less data transmitted over the network

Challenges

- Dynamic tracking of disk chunks at client side
- Storing and caching golden/delta images
- Fast data transmission between Back end Storage, Cache, VM Pool and Frisbee Server
- VM Pool management and fast turn around time for server requests
- Tracking dependencies between images
- Is a block-level representation of data the best choice?
- Security concerns

This material is based upon work supported by National Science Foundation under Grant No. 0709427. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation